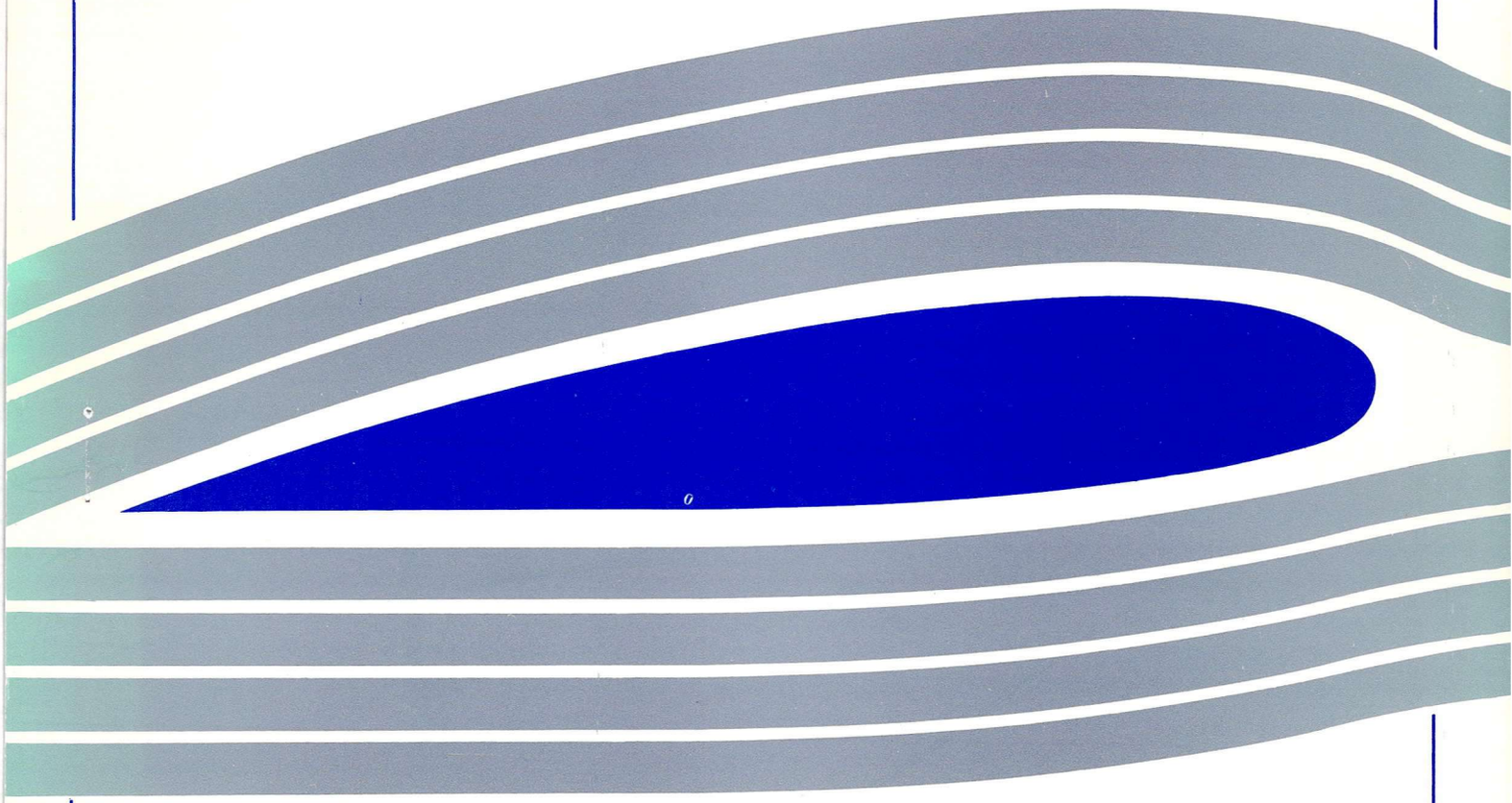
University of Glasgow
DEPARTMENT OF
AEROSPACE
ENGINEERING

Derivation of Analytical Jacobian Matrix for Newton's Method to 3D N-S Equations

Xiaoshi Jin

GU Aero Report 9318, August, 1993



Derivation of Analytical Jacobian Matrix for Newton's Method to 3D N-S Equations

Xiaoshi Jin

GU Aero Report 9318, August, 1993

CONTENTS

Technical Report:

Derivation of Analytical Jacobian Matrix for Newton's Method to 3D N-S Equations

Xiaoshi Jin
Department of Aerospace Engineering
University of Glasgow
Glasgow, G12 8QQ
August, 1993

1.Introduction

Following some studies on the numerical techniques adopted by the CFD group here, it appears that Newton's method is the fastest convergent method to sufficient accuracy. A recent report by Xu[1] indicates that the non-linear system can be solved by using a very robust preconditioner, and significant acceleration of the convergence of the Newton's solver can be achieved. It is considered to extend this method to three dimensional Navier-Stokes solver which will be more useful and comprehensive.

However, there are still some problems to overcome in the application of the Newton's method to three dimensional flows. It is not economic in terms of computer resources used for this method, in particular the computer memory required, which puts the application of it for a fully three dimensional flow in a state of waiting for better hardware development. In a straightforward estimate, a fully three dimensional N-S equations may require 500-1000 Mbytes if Newton's method is implemented, whereas it requires only 25-50Mbytes by using an explicit method. A solution to the problem at the moment is clearly to resort parallel computing, but even though it needs no less than 16 processors with 64Mbytes each or 32 processors with 32Mbytes each in memory. Large amount of computer resources is inevitable.

In additional to the memory requirement, a considerable computing time consumed in evaluating the Jacobian matrix makes the Newton's method less competitive for a big problem such as three dimensional flows. Although Xu's new method[2], which is based on fluxes rather than residuals for numerical approximation of the Jacobian matrix, has significantly reduced the CPU time consumed in forming the matrix, about 20 times of the CPU time used for an explicit iteration is still needed for generating the Jacobian.

In the light of a recent report by Badcock[3], however, it should be noted that the CPU time for the Jacobian matrix can be reduced considerably by using analytical derivatives of the inviscid fluxes. Although only the MUSCL interpolation part of the scheme was actually done in its analytical form in his work, the idea of using analytical derivatives of fluxes gives rise to further feasibility of the application in three dimensional flows.

In this technical report, we give a fully three dimensional analytical Jacobian matrix which is based on the analytical derivatives of Osher's approximate Riemann solver, as well as those of the numerical approximation of the viscous fluxes, and estimate the computer time consumed in forming this kind of Jacobian matrix. The Newton's method, due to the requirement of computer memory, is firstly applied on a three dimensional Parabolized Navier-Stokes (PNS) solver rather than a fully three dimensional case, and all the discussion is based on cell centred finite volume method of a structure grid. A symbolic manipulation package called REDUCE has been used to produce the derivations.

2.Newton's Method

The Newton's method can be written in the following general form:

$$\left(\frac{\partial R}{\partial V}\right)^k \Delta V^k = -R(V^k)$$
$$\Delta V^k = V^{k+1} - V^k$$

where R is the integration residual vector of all cells of a given grid, and V is the solution-to-be vector, and k can be seen as an iteration index.

For the present case R is the residual of integrated Navier-Stokes equations over each cell, and V is the primitive variable vector of the cell.

3.The Jacobian Matrix

The Jacobian matrix is actually composed of the derivatives of the residuals of all cells. For each cell, the residual R is in practice calculated by evaluating the fluxes through all the boundary segments of the cell, that is

$$R_{i,j,k} = F_{i+1/2,j,k} - F_{i-1/2,j,k} + F_{i,j+1/2,k} - F_{i,j-1/2,k} + F_{i,j,k+1/2} - F_{i,j,k-1/2}$$

The residual is a function of the primitive variables which vector can be written as

$$V = [\rho \quad u \quad v \quad w \quad T]$$

The topic is to formulate the following derivatives for each I station

$$\begin{array}{ccccc} & & \dfrac{\partial R_{i,j,k}}{\partial V_{i,j,k+2}} & & \\ & \dfrac{\partial R_{i,j,k}}{\partial V_{i,j-1,k+1}} & \dfrac{\partial R_{i,j,k}}{\partial V_{i,j,k+1}} & \dfrac{\partial R_{i,j,k}}{\partial V_{i,j+1,k+1}} & \\ \dfrac{\partial R_{i,j,k}}{\partial V_{i,j-2,k}} & \dfrac{\partial R_{i,j,k}}{\partial V_{i,j-1,k}} & \dfrac{\partial R_{i,j,k}}{\partial V_{i,j,k}} & \dfrac{\partial R_{i,j,k}}{\partial V_{i,j+1,k}} & \dfrac{\partial R_{i,j,k}}{\partial V_{i,j+2,k}} \\ & \dfrac{\partial R_{i,j,k}}{\partial V_{i,j-1,k-1}} & \dfrac{\partial R_{i,j,k}}{\partial V_{i,j,k-1}} & \dfrac{\partial R_{i,j,k}}{\partial V_{i,j+1,k-1}} & \\ & & \dfrac{\partial R_{i,j,k}}{\partial V_{i,j,k-2}} & & \end{array}$$

It can be readily seen from equation (1) that the derivatives of the residual can be directly obtained from the derivatives of the fluxes since

$$\partial R_{i,j,k} = \partial F_{i+1/2,j,k} - \partial F_{i-1/2,j,k} + \partial F_{i,j+1/2,k} - \partial F_{i,j-1/2,k} + \partial F_{i,j,k+1/2} - \partial F_{i,j,k-1/2}$$

In fact, the evaluation of the derivatives can be done during the same time when the fluxes are evaluated. It also saves CPU time because only the fluxes through one side of the finite volume need to be calculated according to the following equalities

$$F_{i,j+1/2,k}\Big|_{i,j,k} = -F_{i,j-1/2,k}\Big|_{i,j+1,k}$$

$$\left.\frac{\partial F_{i,j+1/2,k}}{\partial V_{i,j,k}}\right|_{i,j,k} = -\left.\frac{\partial F_{i,j-1/2,k}}{\partial V_{i,j-1,k}}\right|_{i,j+1,k} \qquad \left.\frac{\partial F_{i,j+1/2,k}}{\partial V_{i,j-1,k}}\right|_{i,j,k} = -\left.\frac{\partial F_{i,j-1/2,k}}{\partial V_{i,j-2,k}}\right|_{i,j+1,k}$$

$$\left.\frac{\partial F_{i,j+1/2,k}}{\partial V_{i,j+1,k}}\right|_{i,j,k} = -\left.\frac{\partial F_{i,j-1/2,k}}{\partial V_{i,j,k}}\right|_{i,j+1,k} ; \quad \left.\frac{\partial F_{i,j+1/2,k}}{\partial V_{i,j,k+1}}\right|_{i,j,k} = -\left.\frac{\partial F_{i,j-1/2,k}}{\partial V_{i,j-1,k+1}}\right|_{i,j+1,k}$$

$$\left.\frac{\partial F_{i,j+1/2,k}}{\partial V_{i,j+2,k}}\right|_{i,j,k} = -\left.\frac{\partial F_{i,j-1/2,k}}{\partial V_{i,j+1,k}}\right|_{i,j+1,k} ; \quad \left.\frac{\partial F_{i,j+1/2,k}}{\partial V_{i,j,k-1}}\right|_{i,j,k} = -\left.\frac{\partial F_{i,j-1/2,k}}{\partial V_{i,j-1,k-1}}\right|_{i,j+1,k}$$

$$\left.\frac{\partial F_{i,j+1/2,k}}{\partial V_{i,j+1,k+1}}\right|_{i,j,k} = -\left.\frac{\partial F_{i,j-1/2,k}}{\partial V_{i,j,k+1}}\right|_{i,j+1,k} ; \quad \left.\frac{\partial F_{i,j+1/2,k}}{\partial V_{i,j+1,k-1}}\right|_{i,j,k} = -\left.\frac{\partial F_{i,j-1/2,k}}{\partial V_{i,j,k-1}}\right|_{i,j+1,k}$$

Therefore the calculations of the derivatives of the fluxes can be carried out in the same way as it for the fluxes.

The number of the above derivatives to be used in the Jacobian matrix depend on the number of the computational dimensions, and the order of the interpolation to be used. For the first order of MUSCL scheme, only five of the components in the middle are involved for inviscid terms. For computing a fully three dimensional Navier-Stokes equations in third order, twenty five components are required for each finite volume. It can be seen that each of the derivatives is a 5x5 sub-matrix, if we denote in general the above derivatives as follow

$$\frac{\partial R}{\partial V} = \begin{bmatrix} \frac{\partial R_\rho}{\partial \rho} & \frac{\partial R_\rho}{\partial u} & \frac{\partial R_\rho}{\partial v} & \frac{\partial R_\rho}{\partial w} & \frac{\partial R_\rho}{\partial T} \\ \frac{\partial R_u}{\partial \rho} & \frac{\partial R_u}{\partial u} & \frac{\partial R_u}{\partial v} & \frac{\partial R_u}{\partial w} & \frac{\partial R_u}{\partial T} \\ \frac{\partial R_v}{\partial \rho} & \frac{\partial R_v}{\partial u} & \frac{\partial R_v}{\partial v} & \frac{\partial R_v}{\partial w} & \frac{\partial R_v}{\partial T} \\ \frac{\partial R_w}{\partial \rho} & \frac{\partial R_w}{\partial u} & \frac{\partial R_w}{\partial v} & \frac{\partial R_w}{\partial w} & \frac{\partial R_w}{\partial T} \\ \frac{\partial R_E}{\partial \rho} & \frac{\partial R_E}{\partial u} & \frac{\partial R_E}{\partial v} & \frac{\partial R_E}{\partial w} & \frac{\partial R_E}{\partial T} \end{bmatrix}$$

Therefore for three dimensional Navier-Stokes equations each finite volume requires 325(PNS) or 625(3DNS) elements for the Jacobian matrix. This is why a huge amount of computer memory is needed for this Newton-like method. The amount of CPU time consumed in the evaluation of the Jacobian matrix is also very large, in a numerical evaluation, about 20 times much of the CPU time as it for an explicit iteration must be consumed.

4.Analytical Derivatives of MUSCL Scheme

For any cell boundary, the inviscid flux F is a function of the primitive variables from both its left and right sides. The left and right variables are determined by interpolations of their neighbouring cells which determine the accuracy of the scheme. In order to give higher order accuracy a κ-parameter family scheme is adopted for the interpolations,e.g.

$$\left.\begin{aligned} V^L &= V_{i,j,k} + \left\{\left(\frac{s}{4}\right)\left[(1-\kappa s)\Delta_- + (1+\kappa s)\Delta_+\right]V\right\}_{i,j,k} \\ V^R &= V_{i+1,j,k} - \left\{\left(\frac{s}{4}\right)\left[(1+\kappa s)\Delta_- + (1-\kappa s)\Delta_+\right]V\right\}_{i+1,j,k} \end{aligned}\right\}$$

where $\Delta+$ and $\Delta-$ denote forward and backward difference operators, respectively, in i direction; and V is the column hypervector of the primitive variables. κ is a parameter which determines the spatial accuracy of the interpolation. The parameter s serves to limit higher-order terms in the interpolation in order to avoid oscillations at discontinuities such as shock waves in the solutions. The limiting is implemented by locally modifying the difference values in the interpolation to ensure monotone interpolation. s expression is given as

$$s = \frac{2\Delta_+ V \Delta_- V + \varepsilon}{(\Delta_+ V)^2 + (\Delta_- V)^2 + \varepsilon}$$

where ε is a small number preventing division by zero in regions of null gradients.

The chain rule of differentiation is used for the derivatives of the MUSCL scheme, that is,

$$\frac{\partial F}{\partial V_{j+1,k}} = \frac{\partial F}{\partial V_L}\frac{\partial V_L}{\partial V_{j+1,k}} + \frac{\partial F}{\partial V_R}\frac{\partial V_R}{\partial V_{j+1,k}} \qquad \frac{\partial F}{\partial V_{j,k}} = \frac{\partial F}{\partial V_L}\frac{\partial V_L}{\partial V_{j,k}} + \frac{\partial F}{\partial V_R}\frac{\partial V_R}{\partial V_{j,k}}$$

$$\frac{\partial F}{\partial V_{j-1,k}} = \frac{\partial F}{\partial V_L}\frac{\partial V_L}{\partial V_{j-1,k}} \qquad \frac{\partial F}{\partial V_{j+2,k}} = \frac{\partial F}{\partial V_R}\frac{\partial V_R}{\partial V_{j+2,k}}$$

The differentiations of s, V_L and V_R are given as follows:

$$\partial s = \frac{2\left[(\Delta_- - s\Delta_+)V_{j,k}\partial\Delta_+ V_{j,k} + (\Delta_+ - s\Delta_-)V_{j,k}\partial\Delta_- V_{j,k}\right]}{\left[\left(\Delta_+ V_{j,k}\right)^2 + \left(\Delta_- V_{j,k}\right)^2 + \varepsilon\right]}$$

$$\partial V_L = \partial V_{j,k} + \frac{\partial s}{4}\left[(1-2ks)\Delta_- + (1+2ks)\Delta_+\right]V_{j,k} + \frac{s}{4}\left[(1-ks)\partial\Delta_- + (1+ks)\partial\Delta_+\right]V_{j,k}$$

$$\partial V_R = \partial V_{j+1,k} - \frac{\partial s}{4}\left[(1+2ks)\Delta_- + (1-2ks)\Delta_+\right]V_{j+1,k} - \frac{s}{4}\left[(1+ks)\partial\Delta_- + (1-ks)\partial\Delta_+^{\bullet}\right]V_{j+1,k}$$

If we denote

$$A = \left[\left(\Delta_+ V_{j,k}\right)^2 + \left(\Delta_- V_{j,k}\right)^2 + \varepsilon\right] \qquad B = \left[(1-2ks)\Delta_- + (1+2ks)\Delta_+\right]V_{j,k}$$

$$A' = \left[\left(\Delta_+ V_{j+1,k}\right)^2 + \left(\Delta_- V_{j+1,k}\right)^2 + \varepsilon\right] \qquad B' = \left[(1-2ks)\Delta_- + (1+2ks)\Delta_+\right]V_{j+1,k}$$

The following six derivatives can then be given

$$\frac{\partial V_L}{\partial V_{j+1,k}} = \frac{B}{2A}(\Delta_- - s\Delta_+)V_{j,k} + \frac{s}{4}(1+ks)$$

$$\frac{\partial V_L}{\partial V_{j-1,k}} = \frac{B}{2A}(s\Delta_- - \Delta_+)V_{j,k} - \frac{s}{4}(1-ks)$$

$$\frac{\partial V_L}{\partial V_{j,k}} = 1 - \frac{\partial V_L}{\partial V_{j+1,k}} - \frac{\partial V_L}{\partial V_{j-1,k}}$$

$$\frac{\partial V_R}{\partial V_{j+2,k}} = \frac{B'}{2A'}(s\Delta_+ - \Delta_-)V_{j+1,k} - \frac{s}{4}(1-ks)$$

$$\frac{\partial V_R}{\partial V_{j,k}} = \frac{B'}{2A'}(\Delta_+ - s\Delta_-)V_{j+1,k} + \frac{s}{4}(1+ks)$$

$$\frac{\partial V_R}{\partial V_{j+1,k}} = 1 - \frac{\partial V_R}{\partial V_{j+2,k}} - \frac{\partial V_R}{\partial V_{j,k}}$$

5.Analytical Derivatives of Osher's Scheme

For an interior control volume, the inviscid flux-vector through each boundary of the control volume can be written as

$$F = \left[\rho v_n \quad \rho(v_n u + T\gamma_6 n_x) \quad \rho(v_n v + T\gamma_6 n_y) \quad \rho(v_n w + T\gamma_6 n_z) \quad \rho v_n(T\gamma_6\gamma_2 + \theta)\right]$$

where v_n is the dot-product of the local velocity vector with the area vector of the boundary. Obviously, the flux-vector is a function of the local primitive variable-vector V which is given in the following:

$$V = [\rho \quad u \quad v \quad w \quad T]$$

The temperature is preferably used rather than the pressure because of better performance with it in three dimensional codes[4]. The direct derivatives with respect to the boundary variable-vector are

$$\begin{bmatrix} v_n & \rho n_x & \rho n_y & \rho n_z & 0 \\ v_n u + T\gamma_6 n_x & \rho(un_x + v_n) & \rho u n_y & \rho u n_z & \rho\gamma_6 n_x \\ v_n v + T\gamma_6 n_y & \rho v n_x & \rho(v n_y + v_n) & \rho v n_z & \rho\gamma_6 n_y \\ v_n w + T\gamma_6 n_z & \rho w n_x & \rho w n_y & \rho(w n_z + v_n) & \rho\gamma_6 n_z \\ v_n(T\gamma_7 + \theta) & \rho[(T\gamma_7 + \theta)n_x + v_n u] & \rho[(T\gamma_7 + \theta)n_y + v_n v] & \rho[(T\gamma_7 + \theta)n_z + v_n w] & \rho\gamma_7 v_n \end{bmatrix}$$

where n_x, n_y and n_z are the three components of the area vector.

From the last section it can be seen that the following derivatives are needed to be formulated:

$$\frac{\partial F}{\partial(V_L, V_R)}$$

from which it can be seen that a differentiable link between the boundary variables V and those in the left and right sides of the boundary has to be supplied. Osher's approximate Riemann solver is such a link that satisfies the requirement.

Again the chain rule of differentiation is used. If we choose the following characteristic variables to be the intermediate variables for the chain derivatives

$$V' = [c \quad ul \quad vl \quad wl \quad z]$$

where c is the speed of sound, ul, vl and wl are the three components of the velocity at the boundary in local coordinate system, z is the unscaled entropy. If we denote c_x, c_y and c_z as the three components of unit outward vector of the boundary, we then have, e.g. c_x is not zero,

$$\frac{\partial V}{\partial V'} = \begin{bmatrix} \frac{2\gamma_4\rho}{c} & 0 & 0 & 0 & -\gamma_4\rho \\ 0 & c_x & -c_y & -c_z & 0 \\ 0 & c_y & \frac{1-c_y^2}{c_x} & -\frac{c_y c_z}{c_x} & 0 \\ 0 & c_z & -\frac{c_y c_z}{c_x} & \frac{1-c_z^2}{c_x} & 0 \\ 2\rho\gamma_4 c & 0 & 0 & 0 & -T\rho\gamma_6\gamma_4 \end{bmatrix}$$

Therefore, the flux derivatives with respect to the characteristic variables are

$$\frac{\partial F}{\partial V'} = \left[\frac{\partial F}{\partial V}\right]\left[\frac{\partial V}{\partial V'}\right]$$

The primitive variables of the left and right hand side of the cell wall are given as followings:

$$V_L = [\rho_L \quad u_L \quad v_L \quad w_L \quad T_L] \qquad V_R = [\rho_R \quad u_R \quad v_R \quad w_R \quad T_R]$$

According to Osher's scheme, the relations between the characteristic variables and the primitive variables from both sides are (apart from the simplest cases such as $V=V_L$):

$$c = \begin{cases} \gamma_5(v_L + \gamma_0 c_L) & (= c_s^0) \\ \frac{(\gamma-1)}{2(1+\alpha)}[v_L - v_R + \gamma_0(c_L + c_R)] & (= c_{13}) \\ \frac{\alpha(\gamma-1)}{2(1+\alpha)}[v_L - v_R + \gamma_0(c_L + c_R)] & (= c_{23}) \\ -\gamma_5(v_R n - \gamma_0 c_R) & (= c_s^1) \end{cases}$$

$$ul = \begin{cases} \gamma_5(v_L + \gamma_0 c_L) & (= u_s^0) \\ \frac{[v_R - \gamma_0 c_R + \alpha(v_L + \gamma_0 c_L)]}{(1+\alpha)} & (= u_H) \\ \gamma_5(v_R - \gamma_0 c_R) & (= u_s^1) \end{cases}$$

$$z = \begin{cases} \mathrm{Log}\left(\dfrac{T_L\gamma_6}{\rho_L^{\gamma-1}}\right) & (= z_0 = z_{13}) \\ \mathrm{Log}\left(\dfrac{T_R\gamma_6}{\rho_R^{\gamma-1}}\right) & (= z_1 = z_{23}) \end{cases}$$

in which υ_L and υ_R are the dot-products of the local velocity with the unit outward vector of the boundary, and c_L and c_R the local speed of sound, in the left and the right sides, respectively. Hereafter we list some useful derivatives:

$$\frac{\partial\alpha}{\partial\rho_L} = \frac{(\gamma-1)\alpha}{2\gamma\rho_L} \quad \frac{\partial\alpha}{\partial T_L} = -\frac{\alpha\gamma_1}{T_L} \quad \frac{\partial\alpha}{\partial\rho_R} = -\frac{(\gamma-1)\alpha}{2\gamma\rho_R} \quad \frac{\partial\alpha}{\partial T_R} = \frac{\alpha\gamma_1}{T_R}$$

$$\frac{\partial c_L}{\partial\rho_L} = 0 \quad \frac{\partial c_L}{\partial T_L} = \frac{c_L}{2T_L} \quad \frac{\partial c_R}{\partial\rho_R} = 0 \quad \frac{\partial c_R}{\partial T_R} = \frac{c_R}{2T_R}$$

$$\frac{\partial u_H}{\partial\rho_L} = \frac{(\phi_L - u_H)}{(1+\alpha)}\frac{\partial\alpha}{\partial\rho_L} = \frac{\alpha c_{13}}{\gamma(1+\alpha)\rho_L}$$

$$\frac{\partial u_H}{\partial T_L} = \frac{(\phi_L - u_H)}{(1+\alpha)}\frac{\partial\alpha}{\partial T_L} - \frac{\alpha\gamma_0}{(1+\alpha)}\frac{\partial c_R}{\partial T_L} = \frac{\alpha\gamma_0(0.5c_L - \gamma_1 c_{13})}{T_L(1+\alpha)}$$

$$\frac{\partial u_H}{\partial\rho_R} = \frac{(\phi_L - u_H)}{(1+\alpha)}\frac{\partial\alpha}{\partial\rho_R} = -\frac{\alpha c_{13}}{\gamma(1+\alpha)\rho_R}$$

$$\frac{\partial u_H}{\partial T_R} = \frac{(\phi_L - u_H)}{(1+\alpha)}\frac{\partial\alpha}{\partial T_R} - \frac{\gamma_0}{(1+\alpha)}\frac{\partial c_R}{\partial T_R} = \frac{\gamma_0(\gamma_1 c_{23} - 0.5c_R)}{T_R(1+\alpha)}$$

$$\frac{\partial c_{13}}{\partial\rho_R} = -\frac{c_{13}}{(1+\alpha)}\frac{\partial\alpha}{\partial\rho_R} = \frac{\gamma_7 c_{23}}{\rho_R(1+\alpha)}$$

$$\frac{\partial c_{13}}{\partial T_R} = -\frac{c_{13}}{(1+\alpha)}\frac{\partial\alpha}{\partial T_R} + \frac{1}{(1+\alpha)}\frac{\partial c_R}{\partial T_R} = -\frac{\alpha\gamma_1 c_{13}}{T_R(1+\alpha)} + \frac{c_R}{2T_R(1+\alpha)} = \frac{(0.5c_R - \gamma_1 c_{23})}{T_R(1+\alpha)}$$

$$\frac{\partial c_{23}}{\partial\rho_R} = c_{13}\frac{\partial\alpha}{\partial\rho_R} + \alpha\frac{\partial c_{13}}{\partial\rho_R} = -\frac{\gamma_7 c_{23}}{\rho_R(1+\alpha)}$$

$$\frac{\partial c_{23}}{\partial T_R} = c_{13}\frac{\partial\alpha}{\partial T_R} + \alpha\frac{\partial c_{13}}{\partial T_R} = \frac{\alpha(\gamma_1 c_{13} + 0.5c_R)}{T_R(1+\alpha)}$$

$$\frac{\partial c_{13}}{\partial\rho_L} = -\frac{c_{13}}{(1+\alpha)}\frac{\partial\alpha}{\partial\rho_L} = -\frac{c_{23}\gamma_7}{\rho_L(1+\alpha)}$$

$$\frac{\partial c_{13}}{\partial T_L} = -\frac{c_{13}}{(1+\alpha)}\frac{\partial\alpha}{\partial T_L} + \frac{1}{(1+\alpha)}\frac{\partial c_L}{\partial T_L} = \frac{\alpha\gamma_1 c_{13}}{T_L(1+\alpha)} + \frac{c_L}{2T_L(1+\alpha)} = \frac{(0.5c_L + \gamma_1 c_{23})}{T_L(1+\alpha)}$$

$$\frac{\partial c_{23}}{\partial \rho_L} = c_{13}\frac{\partial \alpha}{\partial \rho_L} + \alpha\frac{\partial c_{13}}{\partial \rho_L} = \frac{c_{23}\gamma_7(\gamma-1)}{\rho_L(1+\alpha)}$$

$$\frac{\partial c_{23}}{\partial T_L} = c_{13}\frac{\partial \alpha}{\partial T_L} + \alpha\frac{\partial c_{13}}{\partial T_L} = \frac{\alpha(0.5c_L - \gamma_1 c_{13})}{T_L(1+\alpha)}$$

$$\frac{\partial z}{\partial V} = \begin{bmatrix} \frac{1-\gamma}{\rho} & 0 & 0 & 0 & \frac{1}{T} \end{bmatrix}$$

We now can give the matrix multiplication, the left side for instance

$$\frac{\partial F}{\partial V_L} = \begin{bmatrix}\frac{\partial F}{\partial V}\end{bmatrix}\begin{bmatrix}\frac{\partial V}{\partial V'}\end{bmatrix}\begin{bmatrix}\frac{\partial V'}{\partial V_L}\end{bmatrix}$$

The last two matrix's multiplication can be written as

$$\begin{bmatrix} A_{11} & 0 & 0 & 0 & A_{15} \\ 0 & c_x & -c_y & -c_z & 0 \\ 0 & c_y & \frac{1-c_y^2}{c_x} & -\frac{c_y c_z}{c_x} & 0 \\ 0 & c_z & -\frac{c_y c_z}{c_x} & \frac{1-c_z^2}{c_x} & 0 \\ A_{51} & 0 & 0 & 0 & A_{55} \end{bmatrix}\begin{bmatrix} B_{11} & Bc_x & Bc_y & Bc_z & B_{15} \\ B_{21} & Cc_x & Cc_y & Cc_z & B_{25} \\ 0 & -c_y & -c_x & 0 & 0 \\ 0 & -c_z & 0 & c_x & 0 \\ B_{51} & 0 & 0 & 0 & B_{55} \end{bmatrix} =$$

$$\begin{bmatrix} A_{11}B_{11}+A_{15}B_{51} & c_xBA_{11} & c_yBA_{11} & c_zBA_{11} & A_{11}B_{15}+A_{15}B_{55} \\ c_xB_{21} & c_x^2(C-1)+1 & c_xc_y(C-1) & c_xc_z(C-1) & c_xB_{25} \\ c_yB_{21} & c_xc_y(C-1) & c_y^2(C-1)+1 & c_yc_z(C-1) & c_yB_{25} \\ c_zB_{21} & c_xc_z(C-1) & c_zc_y(C-1) & c_z^2(C-1)+1 & c_zB_{25} \\ A_{51}B_{11}+A_{55}B_{51} & c_xBA_{51} & c_yBA_{51} & c_zBA_{51} & A_{51}B_{15}+A_{55}B_{55} \end{bmatrix}$$

where

$$A_{11} = \frac{2\gamma_4\rho}{c} \quad A_{15} = -\gamma_4\rho \quad A_{51} = 2\gamma_4\rho c \quad A_{55} = -T\rho\gamma_6\gamma_4$$

$$B_{11} = \frac{\partial c}{\partial \rho_L} \quad B_{15} = \frac{\partial c}{\partial T_L} \quad B_{21} = \frac{\partial ul}{\partial \rho_L} \quad B_{25} = \frac{\partial ul}{\partial T_L}$$

$$B_{51} = \frac{\partial z}{\partial \rho_L} \quad B_{55} = \frac{\partial z}{\partial T_L} \quad B = \frac{\partial c}{\partial \nu_L} \quad C = \frac{\partial ul}{\partial \nu_L}$$

The resulting matrix is identical because the denominator c_x (or c_y, c_z) has been cancelled out.

6.The Derivatives of Diffusive Terms

The diffusive terms can also be differentiated with respect to the primitives. For example, the differentations of the stress components with respect to the velocity components can be given as followings:

$$\partial\tau_{xx} = \frac{2\mu}{3\mathrm{Re}}\left[2\partial\left(\frac{\partial u}{\partial x}\right) - \partial\left(\frac{\partial v}{\partial y}\right) - \partial\left(\frac{\partial w}{\partial z}\right)\right] \qquad \partial\tau_{xy} = \frac{\mu}{\mathrm{Re}}\left[\partial\left(\frac{\partial u}{\partial y}\right) + \partial\left(\frac{\partial v}{\partial x}\right)\right]$$

$$\partial\tau_{yy} = \frac{2\mu}{3\mathrm{Re}}\left[2\partial\left(\frac{\partial v}{\partial y}\right) - \partial\left(\frac{\partial u}{\partial x}\right) - \partial\left(\frac{\partial w}{\partial z}\right)\right] \qquad \partial\tau_{xz} = \frac{\mu}{\mathrm{Re}}\left[\partial\left(\frac{\partial u}{\partial z}\right) + \partial\left(\frac{\partial w}{\partial x}\right)\right]$$

$$\partial\tau_{zz} = \frac{2\mu}{3\mathrm{Re}}\left[2\partial\left(\frac{\partial w}{\partial z}\right) - \partial\left(\frac{\partial u}{\partial x}\right) - \partial\left(\frac{\partial v}{\partial y}\right)\right] \qquad \partial\tau_{yz} = \frac{\mu}{\mathrm{Re}}\left[\partial\left(\frac{\partial v}{\partial z}\right) + \partial\left(\frac{\partial w}{\partial y}\right)\right]$$

The heat conduction rate is dependent only on the temperature, their differentations are given as

$$\partial q_x = \frac{1}{\gamma_4 M_\infty^2\,\mathrm{Re}}\left(\frac{\mu}{P_r}\right)\partial\left(\frac{\partial T}{\partial x}\right)$$

$$\partial q_y = \frac{1}{\gamma_4 M_\infty^2\,\mathrm{Re}}\left(\frac{\mu}{P_r}\right)\partial\left(\frac{\partial T}{\partial y}\right) \qquad \partial q_z = \frac{1}{\gamma_4 M_\infty^2\,\mathrm{Re}}\left(\frac{\mu}{P_r}\right)\partial\left(\frac{\partial T}{\partial z}\right)$$

The derivatives in terms of the numerical approximation used in the PNS code[4] can be given here

$$\partial\left(\frac{\partial u}{\partial x}\right)_{j+1/2,k} = \frac{\partial u_{j,k}}{4V_{j+1/2,k}}\left(d\zeta_x^R - d\zeta_x^L - 4d\eta_x^D\right) \qquad \partial\left(\frac{\partial u}{\partial x}\right)_{j+1/2,k} = \frac{\partial u_{j+1,k}}{4V_{j+1/2,k}}\left(d\zeta_x^R - d\zeta_x^L + 4d\eta_x^U\right)$$

$$\partial\left(\frac{\partial u}{\partial x}\right)_{j+1/2,k} = \frac{\partial u_{j,k+1}}{4V_{j+1/2,k}}d\zeta_x^R \qquad \partial\left(\frac{\partial u}{\partial x}\right)_{j+1/2,k} = \frac{\partial u_{j+1,k+1}}{4V_{j+1/2,k}}d\zeta_x^R$$

$$\partial\left(\frac{\partial u}{\partial x}\right)_{j+1/2,k} = -\frac{\partial u_{j+1,k-1}}{4V_{j+1/2,k}}d\zeta_x^L \qquad \partial\left(\frac{\partial u}{\partial x}\right)_{j+1/2,k} = -\frac{\partial u_{j,k-1}}{4V_{j+1/2,k}}d\zeta_x^L$$

$$\partial\left(\frac{\partial v}{\partial y}\right)_{j,k+1/2} = \frac{\partial v_{j,k}}{4V_{j,k+1/2}}\left(d\zeta_y^R - d\zeta_y^L - 4d\eta_y^D\right) \qquad \partial\left(\frac{\partial v}{\partial y}\right)_{j,k+1/2} = \frac{\partial v_{j,k+1}}{4V_{j,k+1/2}}\left(d\zeta_y^R - d\zeta_y^L + 4d\eta_y^U\right)$$

$$\partial\left(\frac{\partial v}{\partial y}\right)_{j,k+1/2} = \frac{\partial v_{j+1,k}}{4V_{j,k+1/2}}d\zeta_y^R \qquad \partial\left(\frac{\partial v}{\partial y}\right)_{j,k+1/2} = \frac{\partial v_{j+1,k+1}}{4V_{j,k+1/2}}d\zeta_y^R$$

$$\partial\left(\frac{\partial v}{\partial y}\right)_{j,k+1/2} = -\frac{\partial v_{j-1,k+1}}{4V_{j,k+1/2}}d\zeta_y^L \qquad \partial\left(\frac{\partial v}{\partial y}\right)_{j,k+1/2} = -\frac{\partial v_{j-1,k}}{4V_{j,k+1/2}}d\zeta_y^L$$

where $d\zeta_y^L$ denotes the average y-component of the area vectors from two interfaces of the immediate left cell in ζ direction.

The viscosity is actually a function of the temperature according to Sutherland's law. The derivative can be easily given as follow:

$$\frac{\partial \mu}{\mu \partial T} = \begin{cases} 1 & T \le {}^{120}/_{T_\infty} \\ \dfrac{(T \cdot T_\infty + 330)}{T(T \cdot T_\infty + 110)} & T > {}^{120}/_{T_\infty} \end{cases}$$

The derivative matrices in general form are given as follows

$$\frac{\partial E_v}{\partial V} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & \dfrac{\partial \tau_{xx}}{\partial u} & \dfrac{\partial \tau_{xy}}{\partial u} & \dfrac{\partial \tau_{xz}}{\partial u} & \tau_{xx} + u\dfrac{\partial \tau_{xx}}{\partial u} + v\dfrac{\partial \tau_{xy}}{\partial u} + w\dfrac{\partial \tau_{xz}}{\partial u} \\ 0 & \dfrac{\partial \tau_{xx}}{\partial v} & \dfrac{\partial \tau_{xy}}{\partial v} & 0 & \tau_{xy} + u\dfrac{\partial \tau_{xx}}{\partial v} + v\dfrac{\partial \tau_{xy}}{\partial v} \\ 0 & \dfrac{\partial \tau_{xx}}{\partial w} & 0 & \dfrac{\partial \tau_{xz}}{\partial w} & \tau_{xz} + u\dfrac{\partial \tau_{xx}}{\partial w} + w\dfrac{\partial \tau_{xz}}{\partial w} \\ 0 & \dfrac{\tau_{xx}}{\mu}\dfrac{\partial \mu}{\partial T} & \dfrac{\tau_{xy}}{\mu}\dfrac{\partial \mu}{\partial T} & \dfrac{\tau_{xz}}{\mu}\dfrac{\partial \mu}{\partial T} & \dfrac{\partial q_x}{\partial T} + \dfrac{1}{\mu}\dfrac{\partial \mu}{\partial T}\left(u\tau_{xx} + v\tau_{xy} + w\tau_{xz} + q_x\right) \end{bmatrix}^T$$

$$\frac{\partial F_v}{\partial V} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & \dfrac{\partial \tau_{yx}}{\partial u} & \dfrac{\partial \tau_{yy}}{\partial u} & 0 & \tau_{yx} + u\dfrac{\partial \tau_{yx}}{\partial u} + v\dfrac{\partial \tau_{yy}}{\partial u} \\ 0 & \dfrac{\partial \tau_{yx}}{\partial v} & \dfrac{\partial \tau_{yy}}{\partial v} & \dfrac{\partial \tau_{yz}}{\partial v} & \tau_{yy} + u\dfrac{\partial \tau_{yx}}{\partial v} + v\dfrac{\partial \tau_{yy}}{\partial v} + w\dfrac{\partial \tau_{yz}}{\partial v} \\ 0 & 0 & \dfrac{\partial \tau_{yy}}{\partial w} & \dfrac{\partial \tau_{yz}}{\partial w} & \tau_{yz} + v\dfrac{\partial \tau_{yy}}{\partial w} + w\dfrac{\partial \tau_{yz}}{\partial w} \\ 0 & \dfrac{\tau_{yz}}{\mu}\dfrac{\partial \mu}{\partial T} & \dfrac{\tau_{yy}}{\mu}\dfrac{\partial \mu}{\partial T} & \dfrac{\tau_{yz}}{\mu}\dfrac{\partial \mu}{\partial T} & \dfrac{\partial q_y}{\partial T} + \dfrac{1}{\mu}\dfrac{\partial \mu}{\partial T}\left(u\tau_{yx} + v\tau_{yy} + w\tau_{yz} + q_y\right) \end{bmatrix}^T$$

$$\frac{\partial G_v}{\partial V} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & \dfrac{\partial \tau_{zx}}{\partial u} & 0 & \dfrac{\partial \tau_{zz}}{\partial u} & \tau_{zx} + u\dfrac{\partial \tau_{zx}}{\partial u} + w\dfrac{\partial \tau_{zz}}{\partial u} \\ 0 & 0 & \dfrac{\partial \tau_{zy}}{\partial v} & \dfrac{\partial \tau_{zz}}{\partial v} & \tau_{zy} + v\dfrac{\partial \tau_{zy}}{\partial v} + w\dfrac{\partial \tau_{zz}}{\partial v} \\ 0 & \dfrac{\partial \tau_{zx}}{\partial w} & \dfrac{\partial \tau_{zy}}{\partial w} & \dfrac{\partial \tau_{zz}}{\partial w} & \tau_{zz} + u\dfrac{\partial \tau_{zx}}{\partial w} + v\dfrac{\partial \tau_{zy}}{\partial w} + w\dfrac{\partial \tau_{zz}}{\partial w} \\ 0 & \dfrac{\tau_{zx}}{\mu}\dfrac{\partial \mu}{\partial T} & \dfrac{\tau_{yz}}{\mu}\dfrac{\partial \mu}{\partial T} & \dfrac{\tau_{zz}}{\mu}\dfrac{\partial \mu}{\partial T} & \dfrac{\partial q_z}{\partial T} + \dfrac{1}{\mu}\dfrac{\partial \mu}{\partial T}\left(u\tau_{zx} + v\tau_{zy} + w\tau_{zz} + q_z\right) \end{bmatrix}^T$$

and finally the viscous part of the Jacobian matrix is consituted by these matrices' dot-product with the area vector of the boundary:

$$\frac{\partial R_v^R}{\partial V} = -\left(n_x \frac{\partial E_v}{\partial V} + n_y \frac{\partial F_v}{\partial V} + n_z \frac{\partial G_v}{\partial V}\right)$$

7.Derivatives of the Boundary Variables

The flux derivatives at wall boundary can be done with only two intermediate groups of two variables

$$V_B = [\rho_B \quad P_B] \qquad V' = [c_B \quad z_R]$$

The flux derivatives are with respect to the primitive variables from only one side of the boundary, e.g.

$$V_R = [\rho_R \quad u_R \quad v_R \quad w_R \quad T_R]$$

After similar derivations, we have

$$\frac{\partial V_B}{\partial V'} = \begin{bmatrix} \dfrac{2\gamma_4\rho_B}{c_B} & -\gamma_4\rho_B \\ 2\gamma_4 c_B\rho_B & -\gamma_4 P_B \end{bmatrix}$$

$$\frac{\partial V'}{\partial V_R} = \begin{bmatrix} 0 & -\dfrac{c_x}{2\gamma_4} & -\dfrac{c_y}{2\gamma_4} & -\dfrac{c_z}{2\gamma_4} & \dfrac{c_R}{2T_R} \\ \dfrac{1-\gamma}{\rho_R} & 0 & 0 & 0 & \dfrac{1}{T_R} \end{bmatrix}$$

$$\frac{\partial V_B}{\partial V_R} = \begin{bmatrix} \dfrac{\rho_B\gamma_4(\gamma-1)}{\rho_R} & -\dfrac{\rho_B c_x}{c_B} & -\dfrac{\rho_B c_y}{c_B} & -\dfrac{\rho_B c_z}{c_B} & \dfrac{\rho_B\gamma_4(c_R-c_B)}{T_R c_B} \\ \dfrac{\gamma_4 P_B}{\rho_R}(\gamma-1) & -c_B\rho_B c_x & -c_B\rho_B c_y & -c_B\rho_B c_z & \dfrac{\gamma_4}{T_R}(\rho_B c_B c_R - P_B) \end{bmatrix}$$

$$A_{51} = \frac{\gamma_4 P_B}{\rho_R}(\gamma-1) \qquad C = -c_B\rho_B \qquad A_{55} = \frac{\gamma_4}{T_R}(\rho_B c_B c_R - P_B)$$

$$\frac{\partial F}{\partial V_B} = \begin{bmatrix} 0 & \rho_B n_x & \rho_B n_y & \rho_B n_z & 0 \\ 0 & 0 & 0 & 0 & n_x \\ 0 & 0 & 0 & 0 & n_y \\ 0 & 0 & 0 & 0 & n_z \\ 0 & P_B\gamma_2 n_x & P_B\gamma_2 n_y & P_B\gamma_2 n_z & 0 \end{bmatrix}$$

$$\frac{\partial F}{\partial V_R} = \frac{\partial F}{\partial V_B}\frac{\partial V_B}{\partial V_R} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ n_x A_{51} & Cc_x n_x & Cc_y n_x & Cc_z n_x & n_x A_{55} \\ n_y A_{51} & Cc_x n_y & Cc_y n_y & Cc_z n_y & n_y A_{55} \\ n_z A_{51} & Cc_x n_z & Cc_y n_z & Cc_z n_z & n_z A_{55} \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

8.Concluding Remarks

The analytical Jacobian matrix for Newton's method has been derived with assistance of a symbolic manipulation package called REDUCE. They have been coded into PNS3D code for testing the efficiency and accuracy. A simple test shows that the CPU used for evaluation the analytical Jacobian matrix is only two times more than it is for an explicit iteration, and the evaluation can be performed during the same time as the evaluation of the residuals. For three dimensional flows, the analytical Jacobian matrix will considerably reduce the CPU time, and make the Newton's method more competitive in its three dimensional extension.

PNS3D should be considered first in extending of the Newton's method. The storage used for a grid of 55x65x75 takes about 45Mbytes, which is feasible for its development in a workstation with 64Mbytes.

Nomenclature

α	ratio of local sound speeds[$=c_{23}/c_{13}=\exp((z_1-z_0)/2\gamma)$]
γ	ratio of specific heats
γ_0	$=2/(\gamma-1)$
γ_1	$=1/(2\gamma)$
γ_2	$=\gamma/(\gamma-1)$
γ_3	$=1/\gamma$
γ_4	$=1/(\gamma-1)$
γ_5	$=(\gamma-1)/(\gamma+1)$
γ_6	$=1/(\gamma M_\infty^2)$
γ_7	$=\gamma_6\gamma_2$
ρ	density
θ	$=0.5(u^2+v^2+w^2)$
κ	parameter for spatial accuracy of the interpolation
Φ_L	Riemann invariant from the left side
μ	molecular viscosity
ε	a small number preventing dividing by zero
Δ	finite difference operator
τ_{ij}	stress tensor (i,j=x,y,z)
q_i	heat conduction rate (i=x,y,z)
M_∞	free stream Mach number
Re	Reynolds number
P_r	Prantl number
T	temperature
T_∞	free stream temperature
x,y,z	global coordinate system
ξ,η,ζ	local coordinate system
u,v,w	velocity components in global coordinate system
$V_{i,j,k}$	volume of cell (i,j,k)
E_v,F_v,G_v	viscous fluxes in i,j,k directions, respectively.

References

[1].Xu, X., BILUF: A Preconditioner of Linear Solver in Newton's Method for Solving Steady State Laminar Locally Conical Navier-Stokes Equations, Draft Report of Department of Aerospace Engineering, University of Glasgow, August, 1993

[2].Xu, X. and Richards, B. E., Simplified Procedure for Numerically Approximate Jacobian Matrix Generation in Newton's Method for Solving the Navier-Stokes Equations, Draft Report for Dept. of Aerospace Engineering, University of Glasgow, Dec. 1992

[3].Badcock, K. J., A Parallelisable Partially Implicit Method for Unsteady Viscous Aerofoil Flows, Technical report, G.U. Aero report 9312, 1993

[4].Jin, X., A Three Dimensional Parabolized Navier-Stokes Equations Solver with Turbulence Modelling, G.U. Aero Report 9315, July, 1993